Remarks

Consideration of the application and allowance of all pending claims are respectfully requested. Claims 1-43 are pending.

In the Office Action, dated October 7, 2004, it appears that the amendments included in the Preliminary Amendment filed with the application on March 10, 2004 were not considered. Examiner Mancho confirmed, via telephone on December 29, 2004, that the Preliminary Amendment was not considered. Thus, applicants respectfully request that the remarks provided below, as well as the remarks and the amendment provided in the Preliminary Amendment, be entered and considered. Further, applicants respectfully request that since the Preliminary Amendment was not considered, that if another Office Action is to follow, that it be designated as non-final. Notwithstanding this request, applicants respectfully submit that their invention is patentable over the cited art, as described below.

Claims 1-43 are rejected under 35 U.S.C. 102(b) as being anticipated by Krause (U.S. Patent No. 6,047,323). Applicants respectfully, but most strenuously, traverse this rejection for the reasons below.

Applicants' invention is directed, in one aspect, to updating components of a computing environment in such a way that both the new version of a component associated with a unit of work and the old version of the component associated with that unit of work are concurrently processing the same or different portions of that unit of work. For example, assume there are multiple copies of a program processing a unit of work. Then, assume that one copy is updated to a new version, whereas the other copies remain at the old version, at least temporarily. The new version and the old versions are able to concurrently process one or more portions of the unit of work, since the new version emulates the old version. That is, the new version mimics behavior of the old versions.

In one embodiment, to update the program, the program is stopped, a new version is loaded and the new program is started. During this process, one or more other copies of the program can continue processing one or more portions of the unit of work. Subsequent to starting the new version of the program, the new version joins the copies of the old version in processing one or more portions of the unit of work.

As one particular example, applicants claim a method of updating components in a computing environment (e.g., independent claim 1). The method includes, for instance, updating a component of the computing environment which is associated with at least a portion of a unit of work from one version to another version; and emulating, by the updated component, the one version, while at least one other component of the computing environment associated with the unit of work remains at the one version, wherein the updated component processes at least a portion of the unit of work concurrent to the non-updated component processing at least a portion of the unit of work. Thus, in applicants' claimed invention, an updated version of a component emulates the non-updated version of the component. This allows both components, the updated component and the non-updated component, to concurrently process one or more portions of the unit of work. For instance, assume there are two copies of a program, Program C, and one of those copies is updated to add new functionality, and thus, is referred to Program C'. As claimed by applicants, Program C' emulates Program C. That is, Program C' imitates (see, e.g., Webster's Ninth New Collegiate Dictionary) Program C allowing Program C' to continue processing the unit of work and to ignore its added functionality for the time being. This is very different from the teachings of Krause.

For example, in Krause, there is no discussion, teaching or suggestion of emulation. In particular, there is no discussion, teaching or suggestion of an updated component associated with a unit of work emulating a non-updated component also associated with that unit of work. There is no imitating by an updated component of a non-updated component in Krause.

Support for this rejection is indicated in Col. 7, lines 65 to Col. 8, lines 1-43 and Col. 9, lines 1-30. However, a careful reading of these sections fails to describe, teach or suggest at least applicants' claimed element of emulating, by the updated component, the one version, while at least one other component of the computing environment associated with that unit of work remains at the one version. There is no description in Krause of an updated component emulating (i.e., imitating) a previous version of the component. The cited sections merely describe that the STREAMS design can be independent and can be distributed. However, there is no discussion at all of emulation. There is no discussion that one version of the STREAMS code can be updated to a new version of the code, and that the new version of the code emulates a previous version of the code, while at least one other STREAMS code remains at the previous version. There is no such discussion in Krause. At the most, in Krause, the standard STREAMS code is modified when the drivers are installed (see, e.g., Col. 3, lines 36-38). The code changes are not performed dynamically, while units of work are being processed, but, rather, the STREAMS code is modified and the modified code is installed, instead of the standard STREAMS code, when the operating system is booted. There is no discussion in Krause of using different STREAMS code versions to process a particular unit of work or of emulating by updated STREAMS code an earlier version of STREAMS code. Since this is missing from Krause, applicants respectfully submit that Krause does not anticipate, teach or suggest applicants' claimed invention.

In addition to the above, applicants respectfully submit that there is no description, teaching or suggestion in Krause of emulating, by the updated component, the one version, while at least one other component of the computing environment associated with the unit of work remains at the one version, wherein the updated component processes at least a portion of the unit of work concurrent to the non-updated component processing at least a portion of the unit of work. There is no discussion in Krause of an updated component and a non-updated component concurrently processing at least a portion of a unit of work, as claimed by applicants. Applicants respectfully submit that this claimed element is not described, taught or suggested in Krause, and thus, not anticipated by Krause. Moreover, applicants respectfully submit that the Office Action fails to describe, teach or suggest where such an element is described in Krause. Thus, applicants respectfully submit that the §102 rejection based on Krause for this claimed element is improper.

The dependent claims are patentable for the same reasons as the independent claims, as well as for their own additional features. For example, in dependent claim 5, applicants recite that the updating includes updating a component identifier of the updated component to correspond to the another version. Applicants respectfully submit that there is no description, teaching or suggestion in Krause of updating a component identifier to correspond to another version of the component. While Krause mentions that the STREAMS framework may be modified, (see, e.g., Col. 1, lines 64 – Col. 2, lines 1-6), there is no description that the updated component has a component identifier that is updated to correspond to the new version. This is not described in Krause.

Further, as another example, in dependent claim 6, applicants recite explicit steps to determine, for instance, whether or not emulation should take place. Again, Krause merely states that a component can be modified, but there is no description in Krause that the earlier version of the STREAMS code is to be emulated. Additionally, there is no description of the steps claimed to determine if emulation is to occur. Since one or more of these elements is missing from Krause, applicants respectfully submit that dependent claim 6 is patentable over Krause.

As yet a further example, claim 7 recites that a cluster version identifier is used to represent a lowest running version of the distributed computing environment, and that the updated component emulates that lowest version when any one component has not been updated to a particular version. Again, there is no such description in Krause.

For at least these reasons, applicants respectfully submit that the dependent claims are patentable over Krause. Applicants respectfully request an indication of allowability for all pending claims.

Should the Examiner wish to discuss this case with applicants' attorney, please contact applicants' attorney at the below listed number.

Respectfully submitted,

Blanche E. Schiller

Blanche E. Schiller
Attorney for Applicants
Registration No.: 35,670

Dated: January 7, 2005.
HESLIN ROTHENBERG FARLEY & MESITI P.C.
5 Columbia Circle
Albany, New York 12203-5160
Telephone: (518) 452-5600
Facsimile: (518) 452-5579